Distributed Database Design for Mobile Geographical Applications

(Final version accepted by the Journal of Database Management)

Manhoi Choy Mei-Po Kwan Hong Va Leong [1]

Abstract

Advanced Traveler Information Systems (ATIS) require efficient information retrieval and updating in a dynamic environment at different geographical scales. ATIS applications are useful in yielding a better utilization of the limited costly transportation arteries and providing value-added traveler information. Many ATIS applications are built on the functionalities provided by *Geographical Information Systems* (GIS), which often cannot meet extra requirements like real-time response. We investigate GIS-based systems in ATIS and propose a system architecture based on GIS and distributed database technology. Issues on data modeling, data representation, storage and retrieval, data aggregation, and parallel processing of queries are discussed. This paper introduces a distributed system architecture for ATIS based on recent technology; presents new data models for information representation; proposes data shipping for efficient query processing and function shipping in reducing communication overhead; exploits network of computers for solving complex problems more timely; and incorporates privacy protection for sensitive data.

1 Introduction

Advances in geographical information systems (GIS) have brought new impacts to the design and application of conventional database systems. This is further enhanced by the ability of locating the whereabout of a user by the global tracking technology like the Global Positioning Systems (GPS) (Zito et al., 1995), which is being exploited by vehicle manufacturers, such as the Mercedes-Benz. The two technologies find their applications in the context of Intelligent Transportation Systems (ITS) (Tsao, 1995). Advanced Traveler Information Systems (ATIS) essentially aim at assisting drivers in trip planning and decision making on destination selection, departure time, route choice, congestion avoidance and navigation (Khattak et al., 1993). This has greatly relieved transportation planning from the traditional costly approach of building freeways and various transit routes.

Specific application requirements of ATIS have been quite demanding. In order to provide traffic information useful for people's travel decision, accurate congestion prediction, enroute real-time traffic warning, and alternate routing suggestions are needed. These operations require real-time processing on large data set over a detailed transportation network. GIS, which allow efficient storage, retrieval and manipulation of spatial and aspatial objects, can provide a basis for ATIS.

Most ATIS research has operated on a simplified street network (Emmerink et al., 1995). GIS, on the other hand, provide a realistic representation of environment for querying and processing. Other information useful for

[1] Manhoi Choy's current address is: Department of Computer Science, The Hong Kong University of Science and Technology, Clear Water Bay, Hong Kong. He is supported by grants HKUST6071/97E and DAG96/97.EG25. Mei-Po Kwan's current address is: Department of Geography, Ohio State University, Columbus, OH 43210, USA. Hong Va Leong's current address is: Department of Computing, The Hong Kong Polytechnic University, Hong Kong. Authorship is in alphabetical order.

traveler decision making can be integrated through geo-referencing. GIS are also highly flexible in manipulating spatial objects and distance according to rules, and different scenarios can be simulated to test the "what-if" cases. GIS operations can help to define individuals' accessibility and mobility in space and time(Kwan, 1998b; Kwan & Hong, 1998). It is particularly helpful in modeling human travel behavior in the context of ATIS (Yim, 1996). Kwan (1998a) has suggested that the requirements of a GIS for modeling human travel behavior in an ATIS context. The system needs to handle not only route information, but also a large amount of explicit spatial information about activity locations in the environment. Both activity locations in the objective and subjective environment (preferred activity locations, preferred routes and familiar areas) would be important in modeling travel behavior. A GIS-interfaced computational process model (CPM) called GISICAS is developed and implemented for activity scheduling behavior in ATIS (Kwan, 1998a). GISICAS contains a comprehensive geographic database in ARC/INFO format to supply detailed spatial information about the potential activity opportunities. It models how the traveler processes the spatial information supplied and makes complex decisions in real-time using spatial search algorithms. The spatial information is stored in a comprehensive geographic database containing a detailed street topological network and activity locations.

The commonly used GIS data models, however, are not without problems. For example, the raster data model divides space into regularly shaped and sized pixels, whereas the topological data model subdivides space into irregularly shaped regions, links and nodes (Frank, 1992). None of them, however, represents traffic movement and interaction very well and the problem of connectivity is not taken into account. Although some GIS packages such as TransCAD (Caliper Corportaion, 1996) and ARC/INFO (ESRI, 1996) implement transportation functions like routing algorithms in the topological data model, they are not without problems. First, the link-node structure is basically planar and would not distinguish an intersection with an overpass which does not cross at grade. This would induce problems for routing unless additional structures in the data model are added. Second, the topological model does not replicate how human perceives the street network. We usually do not think of the street network as segments of links with intersection, but more as the street as a whole. As such, the topological data model is not a natural navigable database (Kwan et al., 1998b). In addition, ITS applications require operations at both regional level and local level. Information may need to be transmitted between different levels of modeling. Mainguenaud (1995) used an object-oriented (OO) approach with graph theory to show how hierarchical relationships can be incorporated into networks. His approach allows single nodes (Master_nodes) or lines (Master_edges) to represent abstractions of sub-networks of the original network. These Master_nodes/edges would thus represent subnodes and edges at a different scale. Although this approach is very useful, dynamic information needed in ITS is not dealt with.

OO data modeling as an alternative for spatial databases was discussed by Worboys et al. (1990), Herring (1992), Kwan et al. (1998b) and Kwan & Speigle (1997). Gahegan & Roberts (1988) suggested an intelligent, OO GIS, which is concerned with increasing the functionality and efficiency of the object-centered approach, and hence increasing the effectiveness of GIS as an aid to analysis and decision-making (Kwan et al., 1998a). It is more natural to treat street networks as different classes of objects that we perceive in the real world. An OO data structure also allows the distinction of an intersection and an overpass due to their membership in different classes. For the micro-macro modeling problem, the use of an OO model facilitates the introduction of new

classes across different levels as well as the introduction of new functions for these classes through inheritance, encapsulation and polymorphism. This may provide a better interface to users and enhance multi-level spatial modeling.

For ATIS applications in which large volume of data and real-time response for accessing data are the critical issues, an acceptable performance can be achieved through the use of spatial indexing and clustering. Many schemes have been proposed including point and regional quadtrees for spatial data partitioning and indexing (Samet, 1989), and R-trees (Guttman, 1984). With respect to object-oriented database (OODB) research, efforts are also dedicated to indexing and related issues. Cobb et al. (1995) developed a self-adjusting indexing scheme for vector data product. Stefanakis and Sellis (1998) dealt with spatial access methods to enhance spatial operations in database management systems. Based on the Morton code sequence and R-tree, Nickerson and Gao (1998) introduced a new hierarchical data structure that supports efficient insertion, deletion and two-dimensional range query operations on swath data. These schemes avoid a serial search of the entire database when handling spatial queries.

In view of the demanding computation for answering queries in ATIS, especially routing problems like the traveling salesperson problem (TSP), parallel computing is regarded as one solution. Chang et al. (1993) presented a traffic network simulation model for real-time applications in ATIS. The proposed simulation model is implemented on a parallel computer for an efficient cost/performance ratio. Their model is implemented with a parallel data structure design and a parallel logic. Preliminary research results show that the running time varies with different levels of model complexities but the parallel simulation methodologies offer a promising alternative in implementing real-time ATIS applications. Furthermore, Imielinski and Badrinath (1992) discussed the use of mobile computers in distributed systems. It is noted that the use of mobile computers in the area of ATIS has not been investigated to a full extent. To utilize the mobile computers owned by users effectively, the problem of communication should be addressed, since the bandwidth of a wireless communication channel is in general quite low, ranging from 19.2kbps per channel upstream to 2Mbps per channel downstream. It is therefore necessary to utilize wisely the scarce wireless bandwidth. The two paradigms for communication in a mobile environment include point-to-point and broadcast. Whenever possible, the more scalable and efficient broadcast paradigm should be employed for a collection of mobile clients. We proposed function shipping to address this bandwidth problem.

In ATIS, information provided to travelers may be affected by decisions made by others in the system. Interrelated decisions for pre-trip planners include the decisions by household members. For enroute travelers, decisions made by other drivers in the system would affect predicted traffic conditions. As a result, some form of consistency control is needed. Kaysi et al. (1993) suggested a consistency check in the system design. However, the consistency issue is not directly dealt with in the database design. In addition to the quality of traffic information provided to travelers, the assurance of privacy is also important and should be integrated in the design of the database.

This paper aims at developing a comprehensive GIS-based system to handle the data representation and data modeling problems for applications in ATIS. Spatial and temporal data aggregations are discussed. The technologies of parallel processing and mobile computing are used. Finally, concurrency control and privacy issues

are incorporated.

The design of our system is application-specific and is targeted on ATIS users. Applications include congestion prediction and routing for pre-trip planners, enroute travelers and emergency vehicles. Data are collected constantly and used for statistical purposes in research on travel behaviors and patterns. These data can prove to be useful for planning purposes as well. In addition, information on locations such as tourist attractions, restaurants, and hospitals is geo-referenced in our GIS-based system. As a result, value-added information like yellow page information, tourist information and suggested shopping and dinning places is readily available to the users.

The main contributions of this paper include: the proposition of a hierarchical distributed system architecture for ATIS based on recent advances in communication networks, database, and distributed system design; the presentation of new data models for the representation of information in ATIS (and other components in ITS) capturing the OO characteristics, the relational properties, and the temporal variations of data; the introduction of data shipping in processing local queries as a means to optimize response time and improve the overall system performance, as well as function shipping in reducing channel contention and communication overhead; the exploitation of a network of distributed computers (or a parallel computer) to solve complex problems such as TSP for better responsiveness; the provision of a simulation study on evaluating the performance of our proposed architecture for query processing with aggregated data; and the incorporation of privacy protection for sensitive data.

The organization of this paper is as follows. In Section 2 we give an overview of our distributed system architecture. In Section 3, we analyze the characteristics of information to be stored in the system and design data models for this information. In Section 4, we discuss the types of queries that our system serves and how they can be handled in our architecture using the data models in Section 3. In Section 5, we perform a simulation study to evaluate the performance of our proposed system. In Section 6, we consider the problem of privacy protection on sensitive data. We conclude with a brief discussion in Section 7.

2 System Architecture

Figure 1 shows an overview of our system. A central site is installed with a set of workstations and some high performance machines and servers, connected by a local area network. A distributed database is maintained among these workstations. A set of base stations is distributed throughout the different regions served by the system. Each base station is installed with one or more low cost computers, with some information stored in the central site replicated locally, including some useful aggregated information. Between the central site and the base stations, a hierarchical structure of computer systems is responsible for managing the computational requests within the region it represents and is interfacing directly with the non-mobile users. Each of these computer systems is installed with several low to medium cost computers. We call these the regional sites.

The central site in the hierarchical structure processes global queries requiring global information, submitted from mobile users. The base stations are responsible for handling some of the global queries that can be solved with aggregated or summary information available at the base stations and for receiving requests and transmitting responses to mobile users in the district covered by the base stations. Depending on the workload of base stations,

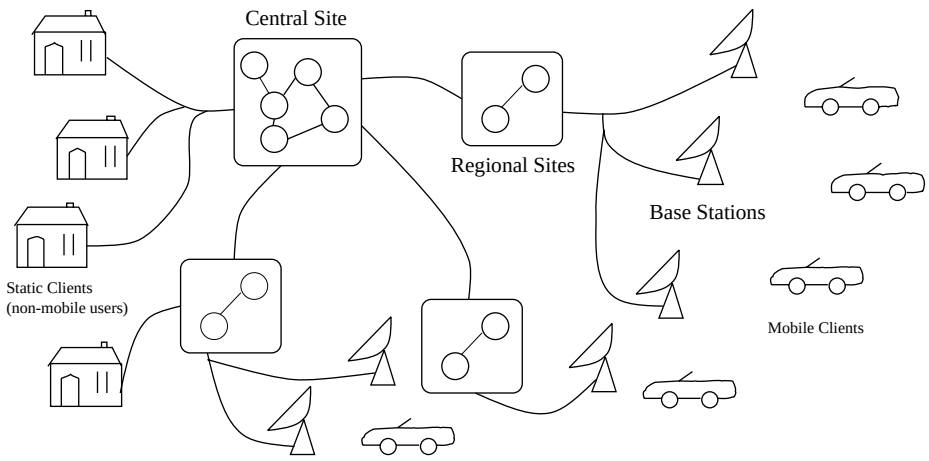

Figure 1: An Overview of the System

they may be equipped with computing units of different computational power. When a base station is overloaded, requests are forwarded to the central site. It is assumed that the central site has enough computational power to handle requests forwarded from the base stations via the intermediate regional sites, which would have processed a certain amount of global queries.

Since mobile users cannot be physically connected to a fixed station, communication between base stations and mobile users is primarily based on a broadcasting medium, such as one using existing microwave broadcasting technology. This communication network can be superimposed on current cellular phone networks. The number of point-to-point uplink channels would be limited. On the other hand, non-mobile users may select to communicate with the central site directly or to their corresponding regional site through telephone lines, Integrated Service Data Network (ISDN) or the cost effective cable modems. Communication between base stations, regional sites and the central site is based on another connection network, perhaps a proprietary network. In small systems, this network may be embedded in the existing telephone system. However, in large systems, this network should be built on high bandwidth medium such as fiber-optics, using the Asynchronous Transfer Mode (ATM) technology (Kim & Wang, 1995) or the Gigabit Ethernet technology (Thompson, 1997).

3 Data Modeling and Representation

Information in our system is classified as either *static* or *dynamic*. Information is classified as static if it remains unchanged over long periods of time and is classified as dynamic if it is updated frequently. For example, road maps, locations of stores, police stations, hospitals, etc., are classified as static information. Traffic conditions such as congestion levels, weather conditions, and the occurrence of accidents are classified as dynamic information. Different ways to maintain these two classes of information are needed in order to maximize the effectiveness of the system and to minimize the amount of data storage required. Two different data models and a combination of approaches for data storage are used here. For static information, a *relational object-oriented* model based on relational database techniques, attribute-list, and the idea of objects is designed while for dynamic information, a *temporal relational object-oriented* model based on a set of *temporal functions*, relational database techniques,

attribute-list, and the idea of objects is developed. These are discussed separately in the following subsections.

3.1 Static Information and the Relational Object-oriented Model

In general, static information is initialized when the system is first set up and is rarely changed during the execution of the system (which may last for years). Even when there is a need to update the static information, it is assumed that relatively slow updating is tolerable. As a result, it is rather straight forward to replicate static information and to propagate updates gradually from central site towards base stations.

Most of the information and requests in our system are location-dependent (Imielinski & Badrinath, 1992). Furthermore, the relative positions between objects, the shape of objects, and a sense of locality are all important factors in processing requests. Consequently, data are modeled as a collection of objects sited on a multi-dimensional plane and the most natural approach is to use OO modeling. However, if an OO model is used in our system, the classification of objects by their relations usually requires the introduction of classes. The definitions of classes in OO models are usually considered static and are not flexible in capturing complex and irregular relationships between objects. Evolution of classes is difficult, if not at all impossible. Moreover, in a pure OO model, there is no systematic way to select and manipulate objects of a certain subclass.

On the other hand, in our *relational object-oriented* model, the way related objects are stored in the system is emphasized. In general, related objects are stored together so that they can be retrieved and updated more efficiently and proper indices are maintained among the related objects. Each object is characterized by a list of attributes. While the extent these attributes cover depends on the size of the system and the services the system is providing, the basic set of attributes should be sufficient to describe the appearance and the orientation of the objects. In the following subsections, we discuss the OO aspect of our system and, in particular, the data representation of three different types of objects, namely point objects, line objects, and area objects. Then, we discuss the relational aspect and the use of relational indices and data aggregation.

3.1.1 Different Types of Objects

Point objects are those that are relatively small and do not extend to cover a significant area. Examples include buildings, police stations, hospitals, stores, and road-side radar detectors. A point object can be simply represented by its relative location and a list of attributes. Line objects include roads, railways, and rivers. Each of them can be represented by a sequence of line segments, which when connected together gives a good approximation of the object. Attributes are also associated with each line segment of the line objects. Area objects can be represented by a variety of methods. A small area object can be represented by a collection of point objects (note that under our definition, although point objects are small, they occupy a non-zero area). A regularly shaped object can be represented by a center point together with the size and shape of the object. A large or irregularly shaped object can be represented by a line object representing the boundary of the area object together with a direction bit, which is used to distinguish between the inside and the outside of the boundary. Moreover, these methods for the representation of area objects can be combined so that different parts of the object are represented using different methods.

3.1.2 Relational Indices and Information Aggregation

To maintain a sense of locality based on the relative locations of objects, information of the entire system is partitioned into *regions* and information within a region is partitioned into *districts*. The hierarchical structuring of information may be made with respect to the physical hierarchical architecture of the ATIS, as depicted in Figure 1. For large systems, districts may also be further partitioned into sub-districts. For example, a region may represent an entire state like Ohio, a district may represent a county such as Franklin County, and a sub-district may represent a city like Columbus. Regions are represented by their locations and the districts of which they are composed. Districts are represented by their relative location in the regions and the collection of objects in the districts. A number of coordinates are used to represent the locations of objects. The principal coordinates determine the region and the district in which an object appears and the remaining coordinates (called the relative coordinates) determine the relative location of the object in its district. Objects that are close to each other, i.e., within the same district (or sub-district if sub-districts are defined), are stored in closely related physical storages so that they can be retrieved and updated in a more efficient manner. Simple queries that only require object-wise information can be served by retrieving data directly from the corresponding district(s) and will not be discussed any more. More complicated queries may require filtering and processing of the information obtained from the districts. These ideas are discussed next.

Retrieving large volume of data for an entire district is costly. To address this issue, the idea of *information aggregation* in the form of data warehouses (Agrawal et al., 1997; Lee et al., 1998; Widom, 1995) is used.

As an example, consider the problem of retrieving population information in a certain region. Instead of returning the population of every location representable in the system, the population of larger area units is returned. Aggregation on static information is stored in a precomputed format, in the form of data cubes (Mumick et al., 1997). Approximated information may even be stored as the space efficient quasi data cubes (Barbara & Sullivan, 1997), where an approximation function on summarized data values, such as mean, second degree moment, etc., are stored instead of the actual data. This saves the scarce network bandwidth between the moving client or vehicle and the base stations.

Information aggregation can be further incorporated with relational database techniques, which provide an efficient indexing mechanism. In our system, relational algebraic operations including *selection, projection, join, union*, and *difference* are supported and indices are built to handle commonly raised queries. These five operations are complete in formulating relational algebra queries (Silberschatz et al., 1996). The operations and indices allow efficient retrieval of related objects and are used to filter out unrelated objects. As an example, consider the following heuristic to approximate the shortest path between two locations in two different districts. We select and extract (using the standard relational operation *selection*) only an outline of the districts along the straight line connecting the two locations. This outline contains main freeways (and highways) in the districts. The retrieval of this outline is performed with the help of an index that links together all the main freeways in the districts. When the appropriate freeway (or highway) entrances and exits are identified, a shortest path can be computed based on the outline. However, some main streets may also be considered in the outline if their inclusion would lead to better connectivity. The criteria under which main streets should also be considered may be either properly set up initially and manually updated periodically or learnt during the execution of the system.

In the latter case, main streets are included in the road map extractions based on dynamic information stored in the system. If a certain main street is being frequently used by users as a connection between freeways, the main street is more likely to be included in the district's road map outline. Dynamic information is discussed in the next section.

3.2 Dynamic Information and the Temporal Relational Object-oriented Model

Dynamic information is collected, analyzed, and stored in the system during normal execution. New information is input frequently and the demand on the availability of the most up-to-date information may be bursty. As new information is input, the problem of maintaining outdated information arises. Since useful information may be deduced from old data, this outdated information should not be discarded completely. To reduce the amount of data storage and yet to be able to retrieve important information, some form of information aggregation is needed. We identify two dimensions of data aggregation, namely *spatial* and *temporal* data aggregation. Spatial data aggregation minimizes the data storage needed for spatially distributed information and reduces the amount of data communications needed for common queries. Temporal data aggregation minimizes the data storage needed for outdated information and reduce the computational requirements for common queries. These two kinds of data aggregation are elaborated next.

To illustrate *spatial* data aggregation of dynamic information, consider the problem of representing congestion levels in the freeway systems. Not every location on freeways representable has an associated value of the congestion level. Instead, congestion levels are sampled at specific locations and averaged over small areas. The congestion level of individual location is obtained by interpolation from congestion levels at nearby locations (along the connecting freeways). Consequently, queries requesting for this information (such as the query requesting for the graphical display of the congestion levels of a certain district) can be served with a smaller amount of data communication. A wide variety of point-based interpolation methods such as moving average, Fourier analysis, and stochastic (Lam, 1993) exist. The selection of the best method for the purpose of interpolating congestion levels is still an unanswered question. Further research in the area of transportation modeling is needed.

To illustrate *temporal* data aggregation of dynamic information, consider the problem of processing a query from a pre-trip planner. A pre-trip planner from Los Angeles visiting Las Vegas by car during the coming weekend would like to find the least congested route. To predict the traffic condition on the way to Las Vegas during the coming weekend, the system needs to know the traffic condition on previous weekends. With temporal data aggregation, only traffic conditions averaged per hourly interval are stored instead of every piece of information available in the previous weekends. The length of intervals that averages are taken may vary depending on the fluctuation of traffic condition. For example, for freeways near cities shorter intervals should be used and for freeways in the deserts longer intervals suffice. Thus, temporal data aggregation limits the amount of outdated data in the system and reduces the amount of computation needed for the request.

Temporal data aggregation is in general much easier to automate compared with spatial data aggregation because of the former's *linear* characteristic. Algorithms that operate on time series can be applicable. In our system, we attempt to incorporate temporal data aggregation systematically and automatically. Given any dynamic information unit, a set of *temporal functions* is provided to retrieve the past history of the unit. The

past history of an information unit may be its statistics, or some averaged value of the unit in a specific time interval. If prediction based on past history is possible for the unit, retrieving *future* (or predicted) value of the unit may also be provided. Depending on the data storage capacity of the system, there may not be enough information stored to return the past history of some information units and, in that case, an appropriate error message is returned. Furthermore, the retrieval of information units based on a set of temporal functions can be extended to a more general fashion. In fact, every query can be composed with any of the temporal functions to yield a new query. For example, the pre-trip planner visiting Las Vegas may issue a query to retrieve the related traffic condition maps during the coming weekend as predicted by the system.

Incorporating temporal relations in the database does not have to be restricted to dynamic information. It is also possible to model all the information in our system with a temporal relational object-oriented model. However, since static information is less likely to be changed over very long period of time, we do not intend to use such a model for static information. Instead, a log keeping the updates to static information should be sufficient. For example, if a user wants to find an older version of a map for a district, he/she is required to go through the updates kept in the log. This is termed a rollback in a temporal database model (Özsoyoğlu & Snodgrass, 1995).

3.3 Data Allocation and Replication

To ensure efficient utilization of data items for computation, especially to improve the response time of accessing remote data items, data should be stored close to where they are consumed. Read-only data should be replicated to allow fast retrieval by a local site, but there is a serious tradeoff between replication and update consistency for mutable data items. Leong & Si (1997) illustrated that redundant information in a mobile environment can be viewed in the form of a cache hierarchy. This hierarchy can be generalized into a hierarchical structure by extending the single database server and replacing it with the hierarchy of base stations, regional sites and central site. We propose here a hierarchical information caching scheme to minimize the data transmission time and access latency, as well as to relieve the problem of channel contention. Caching is a special kind of replication, in which there is a clear master-slave relationship. With a network of distributed computers and abundant storage, the central site maintains all GIS information. Each base station caches the information about its district and aggregated information about neighboring districts. Replicated data in the form of cache are thus partitioned according to the district in the regional sites and the base stations. We will see in the next section that the availability of the cached value, both exact and aggregated, are useful for query processing in the presence of site failures. Mobile clients cache only local information. Through caching, information retrieval time can be greatly curtailed (Barbara & Imielinski, 1994; Si & Leong, 1997; Tan & Yu, 1997).

4 Query Processing and Updating

ATIS should be able to handle user queries from various origins and destinations and at different scales. We classify queries according to the type and amount of information to be retrieved into local queries and global queries. Local queries can be handled efficiently by retrieving a small amount of information and performing

some processing on local computers. Global queries need to access larger amount of information (or aggregated information not available locally), which may be partitioned among several base stations. Base stations are usually mapped to the districts they are located. Global queries may be processed in a base station, or forwarded to the regional sites or the central site for processing.

In the same vein as hierarchical information caching discussed in Section 3.3, we envision a hierarchical query processing structure. Local queries are processed on local computers. Global queries are answered by the computing facilities at the base stations, regional sites, or the central site. Base stations forward queries to the regional sites if they do not have enough information to handle the queries or they are overloaded. Regional sites also forward queries to the central site if they are not capable of answering the queries. In this manner, the workload is distributed among all possible processing agents in the system, creating a higher throughput. In Section 5, some simulation results on the performance of this approach are presented.

Due to the existence of data caching, the system could still deliver partial performance in the presence of central site failure. Since every site will contain information about the districts that it manages, cooperative problem solving may still be possible by requesting the required data from neighboring sites at the same level, though at a high cost of communication. Under this situation, we are forced to perform data shipping from neighboring sites, instead of function shipping to the central site. Direct communication among sites at the same level is initiated. Alternatively, the storage of aggregated information and certain detailed information from neighboring sites allows the production of approximation to certain kind of global queries. For example, the approximated time to pass through a district might be determined from aggregated information of that district. A high approximated value of traveling time will be sufficient to prevent the district from being considered in the shortest path calculation without needing to know the exact value. Whether this approximation is valid depends on the actual application, but the user may be warned about the possibility of getting an inferior result, due to the failure of part of the system.

Taking this further also brings us to a load balancing issue. We can generalize the failure of the central site (or a regional site) into cases when it is overwhelmed with queries during the busy hour. Refusing to execute some of the queries and returning to the regional sites can improve the system performance and response time. Queries can be associated with priorities so that only low priority queries are refused. In this case, refused queries should be solved only based on local information (including aggregated information from other sites) without producing excessive data shipping traffic and yet, approximated solutions can be delivered.

In addition to query processing, the system must also handle the update of information. Static information, as we mentioned earlier, is only updated infrequently and relatively slow update is tolerable. However, dynamic information is updated more frequently and may affect the processing of queries. The results of the queries must observe consistent effects of updates. This brings in the necessity of concurrency control in Section 4.3.

4.1 Local Queries

Local queries involve only local information. A very useful piece of local information is the local map of a district, which can be used to guide the driver through local streets and to search for local facilities such as shopping malls, restaurants and scenic attractions. Other useful information may include indices and details of some of

these facilities. Our focus will be on how to handle the data requirement for these queries, whereas the algorithms are assumed to be off-the-shelf and readily available.

For non-mobile users, local information can be obtained from the central site or the nearest regional site. For mobile users, efficient ways of dispersing local information are needed to process queries locally. Since transmission channels are scarce resources, the simple mechanism of requesting the information on a demand basis from the base station is not only inefficient, but also impossible. This can be verified by a simple calculation on the channel capacity and size of the maps. We use the efficient broadcast mechanism inherent in a base station to transmit the local maps periodically to every user in the district. To further save the communication bandwidth, local maps are transmitted in a compressed form. Any standard compression technique (Ziv & Lempel, 1978) such as LCZ compression suffices for the textual information in the map, as well as relevant indexing information with geo-referencing structures. For a digital map being stored in the bitmap-style image format (for simpler viewer display purpose), more advanced compression techniques like JPEG (ISO 10918, 1991) and even wavelet compression (Schroder et al., 1996) can be utilized. Each user will receive a compressed local map within a short period when the user enters a new district. The local map received is uncompressed, cached and used. This reduces the access latency and, more importantly, channel contention. The broadcast can be arranged carefully to optimize the expected latency (Acharya et al., 1995; Imielinski et al., 1994).

4.2 Global Queries

Global queries need information on the whole region and cannot be solved locally at a client. Therefore they are first forwarded to base stations. With replication of medium scale global information and aggregated information at neighboring base station, base stations can solve most global queries. This is a function shipping strategy rather than data shipping, the strategy adopted in most local queries. We make this design decision to reduce the amount of information to be transmitted over scarce channels to the user computers. Even in the context of data shipping, we prefer the dissemination of essential data via the broadcast paradigm to reduce the strain on the low-bandwidth wireless channels. In this section, we present treatments to the shortest path problem, emergency vehicle routing and the TSP under our framework.

The shortest path problem is a standard problem and there are numerous solutions implemented on various GIS. The solution strategy mentioned earlier in Section 3.1.2 requires the use of outlined information of freeway systems and main streets. Static information of maps, capacity of the individual freeway and freeway system outline are readily available from the databases of the base stations. The dynamic information of traffic congestion includes, for example, traffic volume on each section of the freeways, an exponentially weighed moving average of the number of arriving and departing vehicles from the freeway system, and the effects of particular events, such as the location and number of lanes affected by a road construction or a traffic accident. Base station databases store all information about the local district as well as aggregated information about other neighboring districts. Databases in regional site store information about the several districts it covers as well as aggregated information for database of other regional sites at the same level.

Emergency vehicle routing must be made in reaction to accident and emergency. The system should provide guidelines to allow fast dispatch of service vehicles to the scene of emergency. It determines, on the basis of

the location and type of service vehicles, the vehicles that are capable of carrying out the mission and provides the fastest routes for them. These global queries require further dynamic information such as a profile of the police and emergency vehicles in the district. In collaboration, the system should issue warning messages to other vehicles and turning on the green light along the emergency vehicle route well before its passing, aiming at creating even faster routes for the service vehicles.

A generalization of the TSP is to find a guideline to tour around a city subject to a list of requirements. The TSP is a well-known NP-complete problem. Heuristics are usually adopted to generate solutions that are close to optimal in a reasonable amount of time. Whether TSP is solved by an exact algorithm or a heuristic, it has a nice feature of being divisible into rather independent subproblems. This allows very effective utilization of the distributed computers in the central site, by solving the subproblems on different processors and the speedup is almost linear to the number of processors used. Problems of this nature justify the use of network of inexpensive computers for a cost effective solution.

4.3 Updates and Concurrency Control

As mentioned earlier in Section 3.1, updates to static information occur rather infrequently, and we are concerned mainly with updates to dynamic information. There are various sources of dynamic information. Data collected by various sensor devices located on freeways, information provided by vehicles, information provided by helicopters monitoring traffic flow, and accident reports are examples of sources of dynamic information. Due to their large volume, an efficient way to update dynamic information is needed. Also, due to the concurrent update of dynamic information and query by the user, concurrency control is essential. We describe below the way information is updated in our system and how concurrency control is achieved.

Dynamic information is sent between base stations and the central site in one of the two modes. Under *normal* operation mode, information from base stations is exchanged through the central site via the regional sites periodically. This allows base stations to have an updated view of other base stations. Under *emergency* operation mode, information is exchanged between base stations through the central site immediately. This allows critical information to be available in other base stations as soon as possible. For example, in case a traffic accident occurs in one district, new traffic flow information is immediately transmitted to other base stations. Based on the new traffic flow information, the other base stations may discourage traffic flow towards the district at which the accident occurs and suggest alternative routes.

Next, we consider the issue of concurrency control. The problem of consistency arises when the database is updated in the midst of the processing of a query. Static information updates can be made to appear consistent in an atomic manner with respect to the processing of queries and other updates to the system, by enforcing the rules of concurrency control schemes. This avoids bringing the system to a standstill when a large amount of static information needs to be updated in a consistent manner.

For dynamic information updates, a simple scenario suffices to illustrate the problem of consistency. Consider the occurrence of a traffic accident on a freeway. This information is recorded in the local base station and propagated to the central site and neighboring base stations. Owing to the sudden slowing down of traffic, a lot of drivers may query for an alternative route. Suppose routes α and β are available. When such queries flood

the base station, route α is a better alternative based on the current situation. Without any concurrency control scheme, the answers to all such queries will be α and the net result is that most, if not all, of the vehicles are diverted to route α, creating another congestion as the herd effect. Each answer of α may affect the traffic flow by guiding one more vehicle to route α. In terms of concurrency control theory in multi-user database systems, we should model the query of alternative route as an *update transaction*. The notion of a transaction guarantees that each query will see the effect of another. Even though the queries are executed in parallel, the database behaves *as if* the queries were executed one after another. Sequential execution of queries guarantee correctness, which is summarized as the *ACID* properties (Bernstein et al., 1987). In the above example, after a certain number of queries are executed, the best alternative route may become β.

A commonly used correctness condition for a multi-user database is the *serializability* of transactions (Bernstein et al., 1987). For most updates in an ATIS database, serializability can be enforced by either the *two phase locking* rule, the *time stamp ordering* rule, or even *optimistic concurrency control* (Kung & Robinson, 1981). In the context of the above query/update situation, in which we do not require an absolutely shortest alternative route, it is possible to relax serializability to *bounded ignorance* (Krishnakumar & Bernstein, 1994) or *bounded inconsistency* (Wong & Agrawal, 1992). This allows a greater degree of concurrency and better performance. A variation of such relaxed correctness condition is used in our system. We would like to bound the error E_i in the estimated routing time for each transaction i (or query in our system). This error is defined as the difference between the routing time obtained if the transaction is executed alone and the routing time obtained if the transaction is executed concurrently with other transactions. If the sum of the errors of all the currently active transactions is smaller than a given bound, then the transactions are allowed to execute. Otherwise, the transactions may be blocked or an alternate route may be chosen stochastically among all but the best alternate routes. Generation of a suboptimal answer as well as the termination of a transaction, bring down the sum of the errors. As the sum of the errors is lowered, more transactions can be executed within the error bound.

5 Performance of Hierarchical Aggregation and Query Processing

We conduct a simulation study on the performance of our proposed architecture for query processing involving aggregated data, demonstrating the usefulness of a hierarchical architecture. For simplicity, we consider districts of regular shape and size (in particular, square in our experiment) and investigate into the effect of number of levels in the architecture on the performance. The prime performance metric we measure is the mean delay or response time to handle queries generated by a mobile client in a 24-hour interval. A mobile client will generate a query when its traveling speed is slow, namely, less than 30 miles per hour. We vary the traffic load over a wide range to investigate the impact on the delay. We model the traffic using the common scenario that there are two periods of peak traffic for commuting to work and away from work. The distribution of the road traffic in the 24-hour span is illustrated in Figure 2.

In our simulation, query arrival is modeled as a Poisson process, in which the arrival rate is proportional to the number of vehicles in a particular road segment, i.e., the traffic load. Queries may be requests for information in any district, or at any temporally aggregated interval. Certain queries are solved locally, but some others are

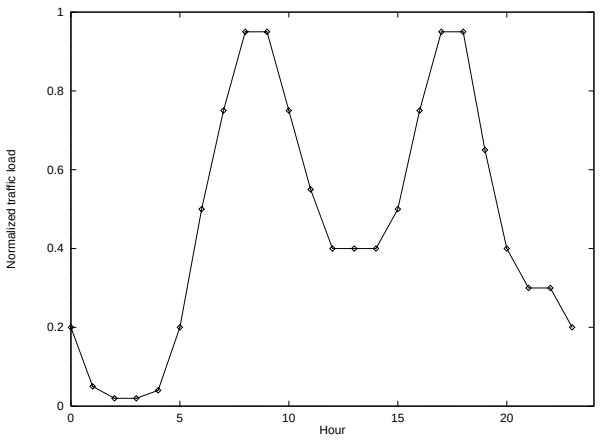

Figure 2: Traffic Loads in Simulated Period

routed to the regional sites and even to the central site for processing. We simulate the multi-lane freeway system of a metropolitan city of size 400 square miles. Incidentally, this is the same as the total area of Hong Kong. We measure the time delay in processing spatial and temporal queries. Our hierarchical architecture has L_m levels. When $L_m = 1$, it is a centralized system with only one single central site, which is also the unique base station. With $L_m = 2$, there is a central site and a number of base stations. With $L_m \geq 3$, there are one or more levels of regional sites in between. For notational convenience, we will denote a site by its level number. The higher level a site is, the larger the area is covered by it. The level number of the central site is always 1 and the level number of a base station is L_m.

The input parameters include the traffic load (measured by the average number of vehicles in a $\frac{1}{4}$-mile, single lane segment of a freeway) and the speed of the vehicles. With the city mentioned above, the resolution is about 80 blocks along each dimension. The traffic information is maintained at one-minute intervals and is temporally aggregated at different levels of details by sites at different levels, k ranging from 1 to L_m. A site at level k manages original data at all levels lower than it, i.e., (temporally non-aggregated) data of the district covered by it as well as data that are temporally aggregated to level k. In a sense, the site has knowledge of all districts covered by it and aggregated information about other districts at the same level which are not covered by it. In our experiments, we vary the number of levels L_m as 2, 3, and 4.

The network is modeled by three major parameters, namely, the network delay, the site delay, and the query delay. The network delay is the delay to transfer a message or request from a mobile client (or a site) to the next site, assumed to be a constant. The site delay is the delay incurred at a site in processing a message due to operating system overhead or queueing delay at the network levels, also assumed as a constant. The query delay is the delay for a site to execute a query using its local database, assumed to follow a normal distribution over an input range. In our simulation, the range of 0.6 to 1.8 seconds is adopted. We measure the final delay that a mobile client will experience.

We divide the queries into categories: a query is said to be at level k if it is intended to be solved at a site at level k, i.e., with aggregated information at that level. We experiment with two different query mixes: (1) even mixture in which each mobile client generates equal amount of queries at any level k and (2) skewed mixture in which the amount of queries at a lower level k is twice the amount of queries at the next higher level $k-1$, forming

a geometric series. We will call the former even query mix and the latter skewed query mix. We believe that the latter mix is more common. There are some queries designated to be answered at level k but subsequently found to require additional information from other sites at the same level. We test both small amount of those "outreaching" queries and moderate amount of them in our experiments, duly adjusted so as to yield comparable results with the value in even and skewed query mixes.

$k \backslash L_m$	2	3	4
1	80	80	80
2	20	30	30
3		15	15
4			8

Table 1: Size of Districts at Different Levels

Since the number of blocks along a direction is not a perfect power, there are some minor adjustments to the size of the districts or regions. We have adopted the size of each district at level k with various values of L_m, as shown in Table 1. With the given district sizes, we perform four experiments and the results are illustrated in Figure 3. Figures 3a and 3b depict results for the even query mix while Figures 3c and 3d depict those for the skewed query mix. Furthermore, Figures 3a and 3c depict results when the amount of out-reaching queries is small while Figures 3b and 3d depict those with more out-reaching queries.

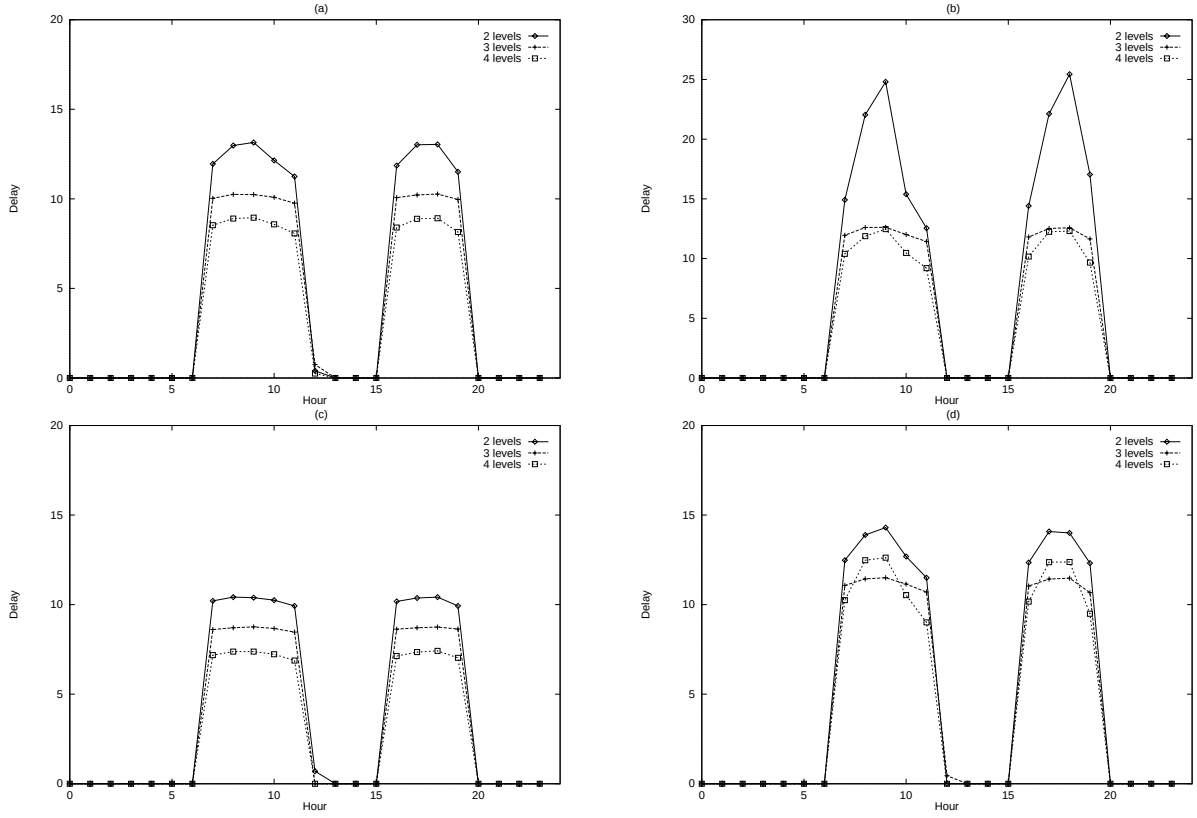

Figure 3: Simulation Results

Each simulated experiment for a 24-hour period is repeated 20 times and we measure the metric and compute the means of the 20 measures. We discover that in most of the cases, the standard deviation is about 0.2 to 0.4 of the value of the mean. This translates into a 95% confidence interval of about ±8 to 17% of the metric. The performance differences in the experiments are therefore significant. From the figure, we observe that the more levels we have, the smaller the delay in the system, but, of course, the more expensive it is. We also observe that with more out-reaching queries, an increase in the number of levels might not be as beneficial as shown in Figure 3*d*. Nevertheless, the improvement from two levels to three levels is rather significant in all cases and it is thus recommended. With a small number of out-reaching queries, the performance is rather stable but it is much affected when the traffic becomes heavier. We thus should encourage the issue of queries that are not out-reaching in nature in the presence of traffic jam. Last, but not least, the response times of the queries are still within acceptable range.

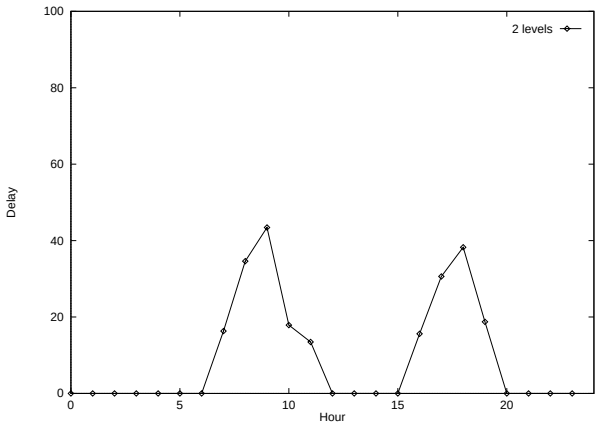

Figure 4: Simulation Results for Central Site Saturation

In the experiments above, the computers at the central site are not saturated. We next perform a small trial run that is dumping more queries on the central site by forwarding more queries to it such that it touches a saturation point when queries begin to hold up and backlog starts to accumulate. We do this for $L_m = 2$ and the result is illustrated in Figure 4. Here we can observe that the response time is already doubled. A slight increase in the query submission will drive up the response time rapidly. This calls for a future research topic in the context of network communication, namely, input flow control.

6 Privacy Protection

Privacy and security are important issues in a public information system. When base stations have access to the information associated with each user's vehicle, the unrestricted disclosure of this information constitutes a violation to the Act of Privacy. Consider a base station that knows the location of a vehicle every now and then. By taking the temporal and spatial information of the vehicle, it may well reveal that the vehicle is traveling at a speed of 80 mph (exceeding the 65 mph limit). Even though the driver of the vehicle is violating the traffic law, he/she will not want this to be disclosed, if he/she is not caught on scene by the police. Similarly, the whereabout of a vehicle may be a piece of sensitive information, especially when this is clearly identified with a time. The

traffic pattern of a particular vehicle is regarded as private information. Mechanisms must be devised to ensure privacy protection. Also, sufficient information must be available to the system, to answer queries.

An easy solution to this is the *anonymous vehicle policy*. The system preserves information about each vehicle, but removes the identification of the vehicle. This anonymous information can then be extracted to generate the aggregated information. Vehicle identifications are only kept during the query computation and destroyed immediately afterwards, leaving only enough information for statistical purposes. To satisfy billing purposes, the originating times of queries and resources spent on their processing must also be recorded. This information must be decoupled from the content of the query and the location from which the query is originated to ensure privacy. For example, information can be decoupled into two relations $statistics = \langle vehicle, time, query\_type, query\_expense \rangle$ and $private = \langle vehicle\_id, location, query\_content \rangle$. The decoupling of information in such a manner is a feasible, but not the best, solution as it becomes impossible resolving certain disputes between the users and the service provider later on. Also, it is not possible for users to present records of their vehicles as evidence of not involving in some criminal acts to law enforcement agency.

Cryptographic techniques can be used to maintain full accounting information without compromising privacy. We propose to employ a public key encryption scheme, such as the RSA scheme (Rivest et al., 1978). There are distinct functions associated with each user: an encryption function $E(P)$ and a decryption function $D(P)$ for a piece of information P. The pair of functions must satisfy $D(E(P)) = E(D(P)) = P$, for any P. It must be impossible to deduce D from E and for any P' very similar to P, $E(P)$ and $E(P')$ must look very different. The function E is known to the public and D is kept secret by the user. The tuples of *statistics* and *private* are indexed by sequence numbers. A relation *connection* is defined to relate the records of *statistics* and the records of *private*. A pair (a, b) is in *connection* if a is the sequence number of a record in *statistics*, b is the sequence number of a record in *private*, and these records are resulted from the same query. Privacy protection of sensitive information is achieved by protecting the disclosure of the relation *connection*. Instead of storing the relation *connection* directly, it is stored as encrypted tuples of the form $\langle s.seq, E(p.seq \cdot D(s.seq)) \rangle$ where $s.seq$ and $p.seq$ are the sequence number of the corresponding *statistics* and *private* records and $\cdot$ represents concatenation. A query is executed as follows: the system generates $s.seq$, the user computes $D(s.seq)$ and returns to the system. The system verifies that the user is not lying when $E(D(s.seq)) = s.seq$. When the query completes, the tuple $\langle s.seq, E(p.seq \cdot D(s.seq)) \rangle$ is generated and stored. It is impossible for the system to fake the tuple, since $D(s.seq)$ cannot be generated without D. It is also impossible for the system to deduce $p.seq$ from $s.seq$ without knowing D. The user can disclose the information by presenting his/her key D to compute $D(E(p.seq \cdot D(s.seq))) = p.seq \cdot D(s.seq)$ and to retrieve the complete record.

The above solution suggests homogeneous recording of all the information of a user – only one pair of keys is used for each user. Exposing the private key of the user for verification of certain information will open the door to accessing *all* records of the user. It is not acceptable in many case for this all-or-none protection. A simple way to get out of this dilemma is to introduce the session key, which is known to both user and the system, for symmetric or private key encryption. The user can use his/her public key and private key to communicate the session key to the system. Each user will select a session key which is valid for a period of time. The session key is maintained by and may be changed occasionally by the client. The system should discard the session key once

it is done with it. To ensure for the validity of session key, timing information is also included. The session key k is used to encrypt the information. Let the encryption function using k be e_k. The final encrypted tuple is stored in the form $\langle s.seq, E(e_k(p.seq \cdot time) \cdot D(s.seq \cdot time))\rangle$. The inclusion of the time component further provides freshness control to guard against stale data items. To disclose the information, both private key and session key must be used.

Despite the cryptographic approach for information protection, the complete system would gather accounting information in an anonymous manner and just maintain the encrypted copies for dispute resolution and other mutually consent usages. Different levels of security and privacy can be defined based on the sensitivity of the data, and different encryption techniques can be employed.

7 Conclusion

In this paper, we have proposed a system architecture composing of a central site, a certain number of regional sites, a set of base stations, non-mobile computers hooked up to the system via phone lines or fiber optics, and mobile computers connected to the system via a broadcasting medium. Several levels of processing powers, from users' personal computers, to base stations' workstations, to regional sites' high-end workstations, and to the central site's high performance distributed computers, contribute to a high degree of concurrency in the system. New data models, the *relational object-oriented* model and the *temporal relational object-oriented* model, are introduced to represent ITS information. These data models allow efficient representation of both static and dynamic information that are inherent to ITS. Information aggregation is supported in both spatial and temporal domains.

However, our model is not without limitations. First, there is a lack of a standard and efficient query language for OODB. More research in this direction is needed. Second, existing limitations on network bandwidth would be a problem for large volume of data transfer. Hopefully, next generation of gigabit networks will alleviate this problem. Third, failure in the central site will deter the performing of global functions in our model. In this situation, at least the system can still partially function, as we have shown in the simulation, and have the work load shared among regional sites to provide perhaps approximate solution to queries.

Although our system is mainly designed for ATIS users, it can be extended to include many other functions. Considerations may be given to the extension of our system for ATIS to a more general database system that provides general information to the public. We have performed some simulated experiments on the response time of queries generated in the system under the traffic loading. More work is needed to investigate the feasibility and the cost effectiveness of such a system.

References

Acharya, S, Franklin, M., & Zdonik, S. (1995). Dissemination-based data delivery using broadcast disks. *IEEE Personal Communications*, 2(6):50–60.

Agrawal, D., El Abbadi, A., Singh, A., & Yurek, T. (1997). Efficient view maintenance at data warehouses. In *Proceedings of the ACM International Conference on Management of Data*, 417–427.

Barbara, D. & Imielinski, T. (1994). Sleepers and workaholics: Caching strategies in mobile environments. In *Proceedings of the ACM International Conference on Management of Data*, 1–12.

Barbara, D. & Sullivan, M. (1997). Quasi-cubes: Exploiting approximations in multidimensional databases. *SIGMOD Record*, 26(3):12–17.

Bernstein, P.A., Hadzilacos, V. & Goodman, N. (1987). *Concurrency Control and Recovery in Database Systems.* Addison Wesley, Reading, Massachusetts.

Caliper Corporation (1996). *Routing and Logistics with TransCAD.*

Chang, G., Junchaya, T. & Santiago, A.J. (1993). A real-time network traffic simulation model for ATMS applications. In *Transportation Research Board 72nd Annual Meeting*, 10–14.

Cobb, M., Chung, M., Shaw K., & Arctur D. (1995). A self-adjusting indexing structure for spatial data. In *GIS/LIS Proceedings*, 182–192.

Emmerink, R.H.M., Axhausen, K.W., Nijkamp, P. & Rietveld, P. (1995). The potential of information provision in a simulated road transport network with non-recurrent congestion. *Transportation Research C*, 3(5):291–309.

ESRI (Environmental Systems Research Institute, Inc.) (1996). *Understanding GIS: The ARC/INFO Method.*

Frank, A.U. (1992). Spatial concepts, geometric data models, and geometric data structures. *Computers and Geosciences*, 18(4):409–418.

Gahegan, M.N. & Roberts, S.A. (1988). An intelligent, object-oriented geographical information system. *International Journal of Geographical Information systems*, 2:101–110.

Guttman, A. (1984). R-trees: A dynamic index structure for spatial searching. In *Proceedings of ACM International Conference on Management of Data*, 47–57.

Herring, J. (1992). TIGRIS: a data model for an object-oriented geographic information system. *Computers and Geosciences*, 18(4):443–452.

Imielinski, T. & Badrinath, B.R. (1992). Querying in highly mobile distributed environment. In *Proceedings of the 18th International Conference on Very Large Databases*, 41–52.

Imielinski, T., Vishwanathan, S. & Badrinath, B.R. (1994). Power efficient filtering of data on the air. In *Proceedings of the 4th International Conference on Extending Database Technology*, LNCS, 245–258.

ISO 10918 (1991). Digital compression and coding of continuous-tone still images. Technical Report ISO 10918, ISO.

Kaysi, I., Ben-Akiva, M. & Koutsopoulos, H. (1993). An integrated approach to vehicle routing and congestion prediction for real-time driver guidance. In *Transportation Research Board 72nd Annual Meeting*, 10–14.

Khattak, A.J., Schofer, J.L. & Koppelman, F.S. (1993) Commuters' enroute diversion and return decisions analysis and implications for advanced traveler information systems. *Transportation Research A*, 27(2):101–111.

Kim, B.G. & Wang P. (1995). ATM networks: goals and challenges. *Communications of the ACM*, 38(2):39–44.

Krishnakumar N. & Bernstein, A.J. (1994). Bounded ignorance: a technique for increasing concurrency in a replicated system. *ACM Transactions on Database Systems*, 19(4):586–625.

Kung, H.T. & Robinson, J.T. (1981). On optimistic methods for concurrency control. *ACM Transactions on Database Systems*, 6(2):213–226.

Kwan, M.-P. (1998a). GISICAS: An activity-based travel decision support system using a GIS interfaced computational-process model. In *Activity-Based Approaches To Travel Analysis*, 263-282.

Kwan, M.-P. (1998b). Space-time and integral measures of individual accessibility: A comparative analysis using a point-based method. *Geographical Analysis*, 30(3):191–216.

Kwan, M.-P., Golledge, R. & Speigle, J. (1998a). Developing an object-oriented datamodel for use in an advanced traveler information system. In *Theoretical Foundations of Travel Choice Modelling.*

Kwan, M.-P., Golledge, R. & Speigle, J. (1998b). A review of object-oriented approaches in geographic information systems for transportation modeling. *Manuscript, the Ohio State University.*

Kwan, M.-P., & Hong, X. (1998). Network-based constraints-oriented choice set formation using GIS. *Geographical Systems*, to appear.

Kwan, M.-P. & Speigle, J. (1997). Developing an object-oriented testbed for modeling transportation networks. *Paper presented at the Western Regional Science Association Annual Meeting, Kona, Hawaii.*

Lam, N. (1993). Spatial interpolation methods: a review. *The American Cartographer*, 10(2):129–149.

Lee, K.C.K., Leong, H.V. & Si, A. (1998). Incremental view update for a mobile data warehouse. In *Proceedings of the 13th ACM Symposium on Applied Computing*, 394–399.

Leong, H.V. & Si, A. (1997). Database caching over the air-storage. *The Computer Journal*, 40(7):401–415.

Mainguenaud, M. (1995). Modelling the network component of geographical information systems. *International Journal of Geographic Information Systems*, 9:575.

Mumick, I.S., Quass, D. & Mumick, B.S. (1997). Maintenance of data cubes and summary tables in a warehouse. In *Proceedings of the ACM International Conference on Management of Data*, 100–111.

Nickerson, B.G. & Gao, F. (1998). Spatial indexing of large volume swath data sets. *International Journal of Geographical Information Systems*, 12(6):537–560.

Özsoyoğlu, G. & Snodgrass, R. (1995). Temporal and real-time databases: A survey. *IEEE Transactions on Knowledge and Data Engineering*, 7(4):513–532.

Rivest, R.L., Shamir, A. & Adleman, L. (1978). A method for obtaining digital signatures and public key cryptosystems. *Communications of the ACM*, 21:120–126.

Samet, H. (1989). *The Design and Analysis of Spatial Data Structures.* Addison Wesley, New York.

Schroder, P., Sweldens, W., Cohen, M., DeRose, T. & Salesin, D. (1996). *Wavelets in computer graphics.* ACM SIGGRAPH 96 Course Notes.

Si, A. & Leong, H.V. (1997). Adaptive caching and refreshing in mobile databases. *Personal Technologies*, 1(3):156–170.

Silberschatz, A., Korth, H.F. & Sudarshan, S. (1996) *Database System Concepts.* McGraw-Hill, New York, third edition.

Stefanakis, E. & Sellis, T. (1998) Enhancing operations with spatial access methods in a database management system for GIS. *Cartography and Geographic Information Systems*, 88(4):270–273.

Tan, K.L. Yu, J.X. (1997). A dynamic scheduler for the infinite air-cache. *Data and Knowledge Engineering*, 24(1):97–112.

Thompson, G.O. (1997). Work progresses on Gigabit Ethernet. *IEEE Computer*, 30(5):95–96.

Tsao, H.S.J. (1995). Traffic control for automated highway systems: A conceptual framework. *Transportation Research C*, 3(4):227–246.

Widom, J. (1995). Research problems in data warehousing. In *Proceedings of the 4th International Conference on Information and Knowledge Management*, 25–30.

Wong, M.H. & Agrawal, D. (1992). Tolerating bounded inconsistency for increasing concurrency in database systems. In *Proceedings of the ACM Symposium on Principles of Database Systems*, 236–245.

Worboys, M.F., Hearnshaw, H.M. & Maguire, D.J. (1990). Object-oriented data modelling for spatial databases. *International Journal of Geographical Information Systems*, 4(4):369–383.

Yim, Y. (1996). The effects of traffic information on traveler behavior. *University of California at Berkeley, ITS/PATH.*

Zito, R., D'este, G. & Taylor, M.A.P.. (1995). Global positioning system in the time domain: How useful a tool for intelligent vehicle highway systems. *Transportation Research C*, 3(4):193–209.

Ziv, J. & Lempel, A. (1978). Compression of individual sequences via variable-rate coding. *IEEE Transactions on Information Theory*, 24(5):530–536.